Video 30

Sunday, November 22, 2020 8:42 PM

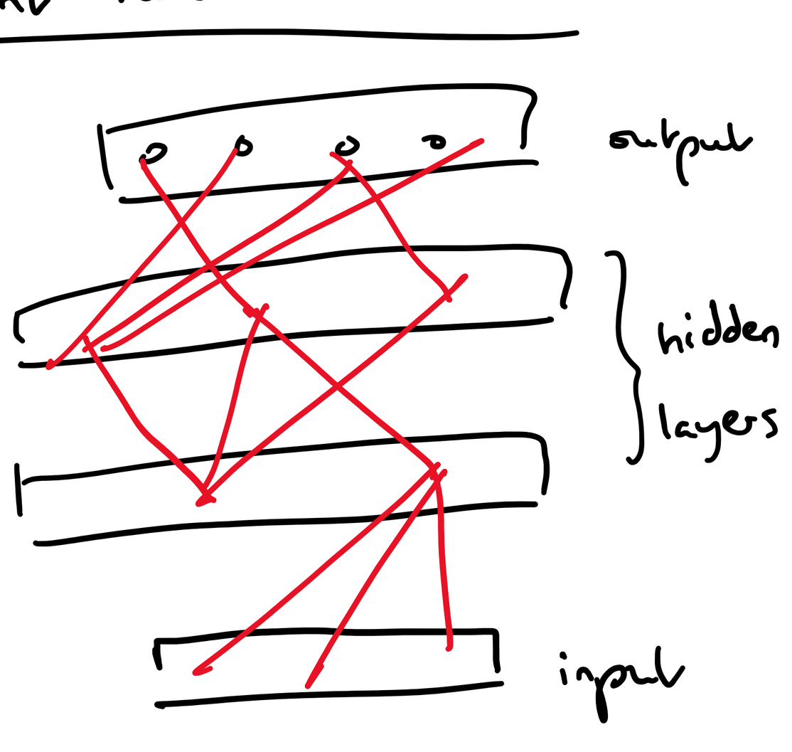
Sequence - to - sequence prediction

Both the input and the output are sequences

Ex applications :

- speech - to - text :
  - input : audio waveform (human speech)
  - output : text that was spoken
  - alternative : text - to - speech
- caption generation :
  - input : image (not necessarily modeled as a sequence)
  - output : natural language description of the image.
- machine translation : seq - to - seq
  - input : sentence in one language (English)
  - output : ______ another language (French)

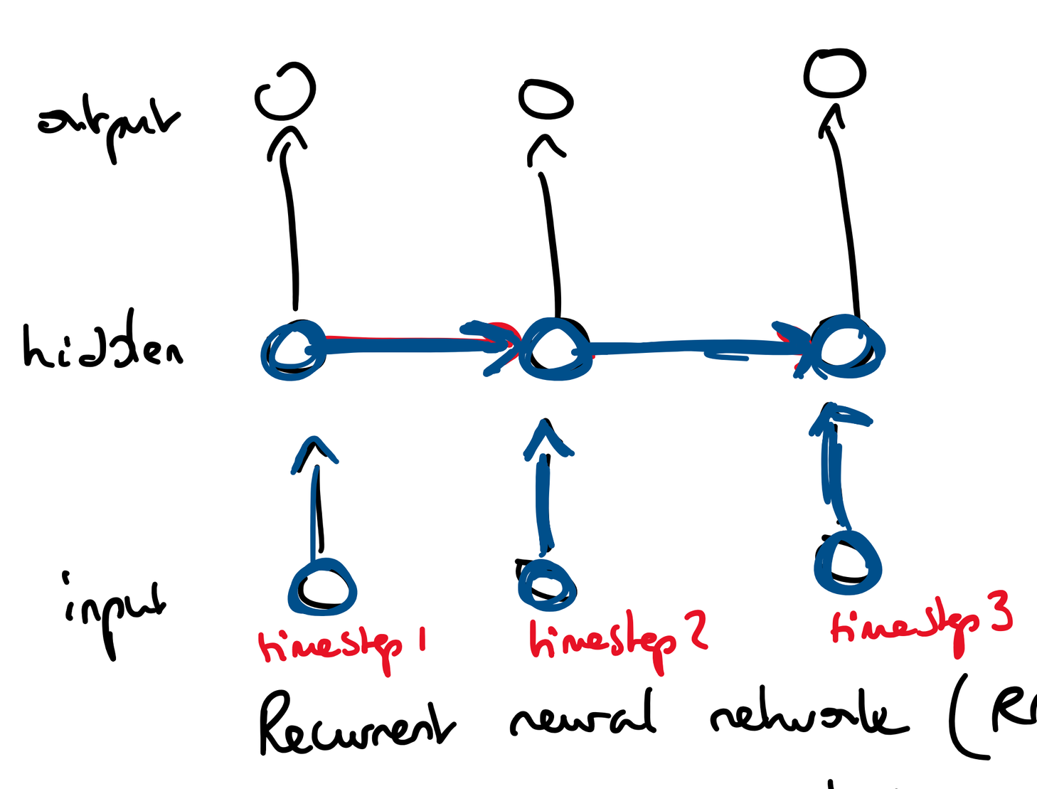
Recurrent neural network



fully connected / convolutional neural network

neurons are only connected to neurons in other layers (preceding, following)

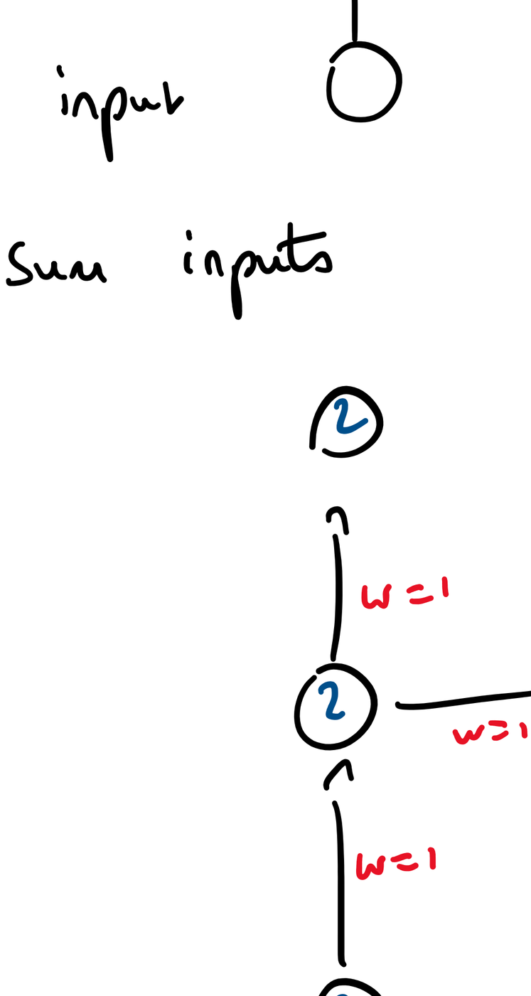
Directed acyclic graphs



Recurrent neural network (RNN) allow connections between neurons of the same layer

Directed graph which may contain cycles

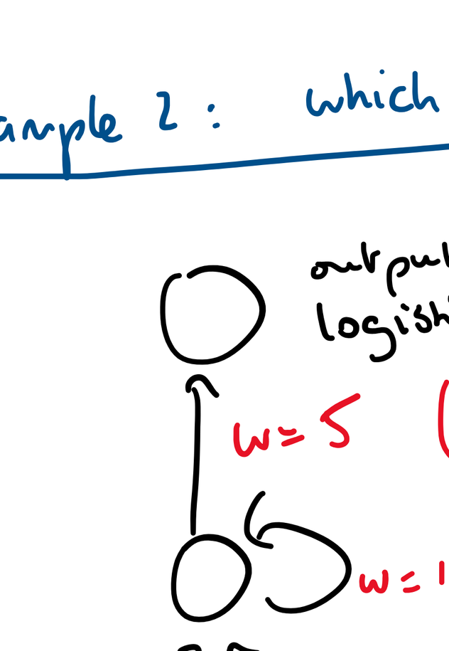
Cycles in RNNs

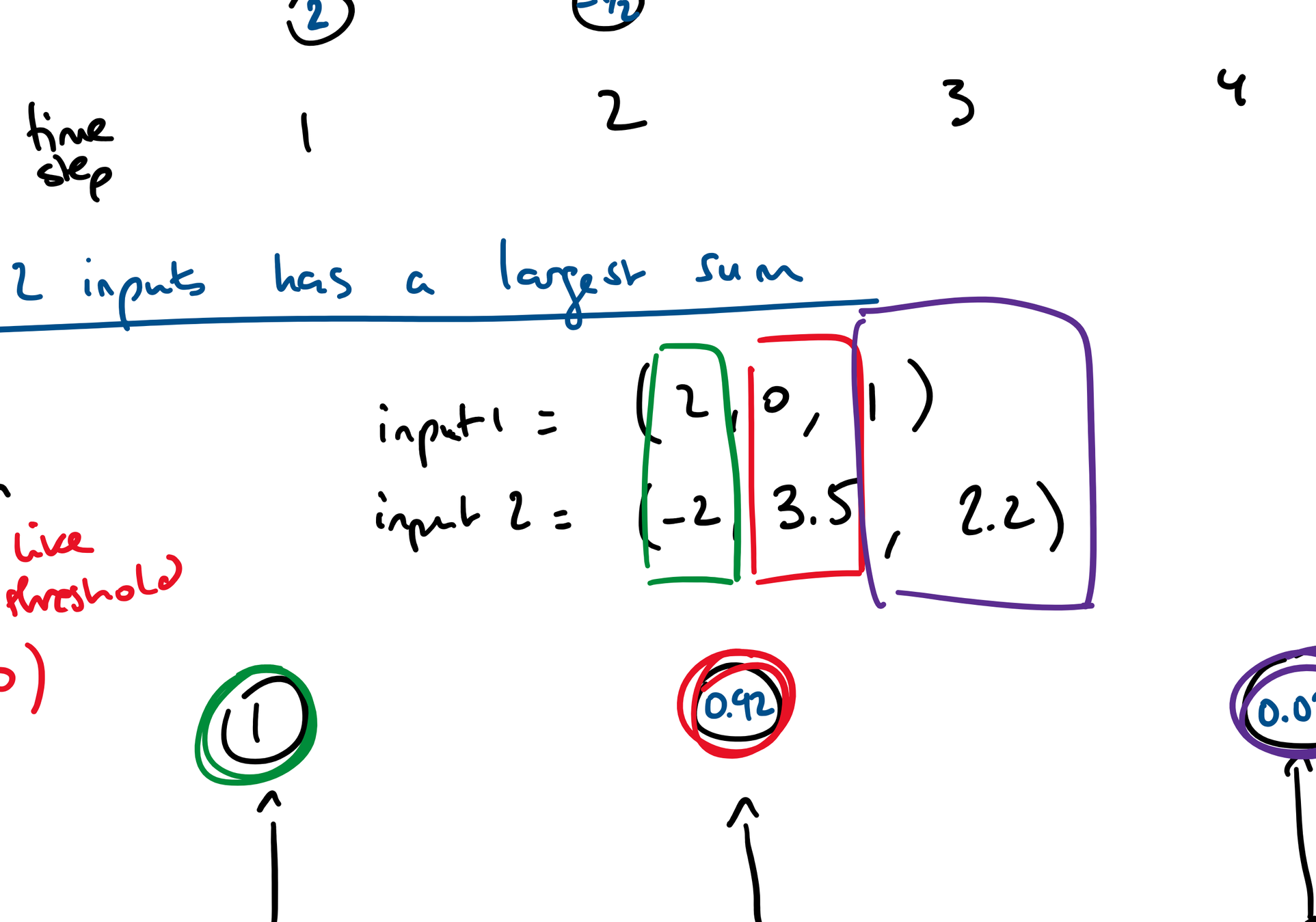


connection connects the hidden neuron to itself at the previous time step

Example : RNN to sum inputs

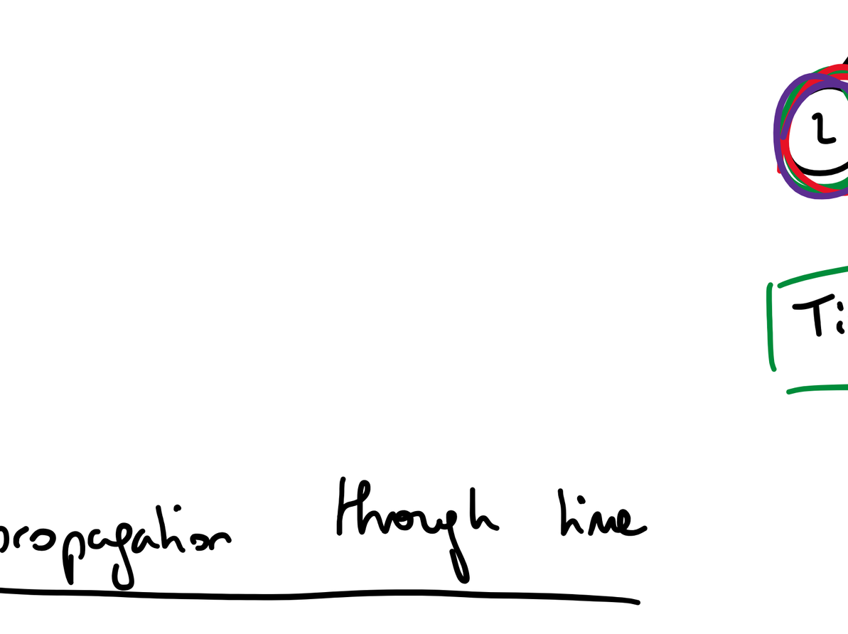
All neurons are linear





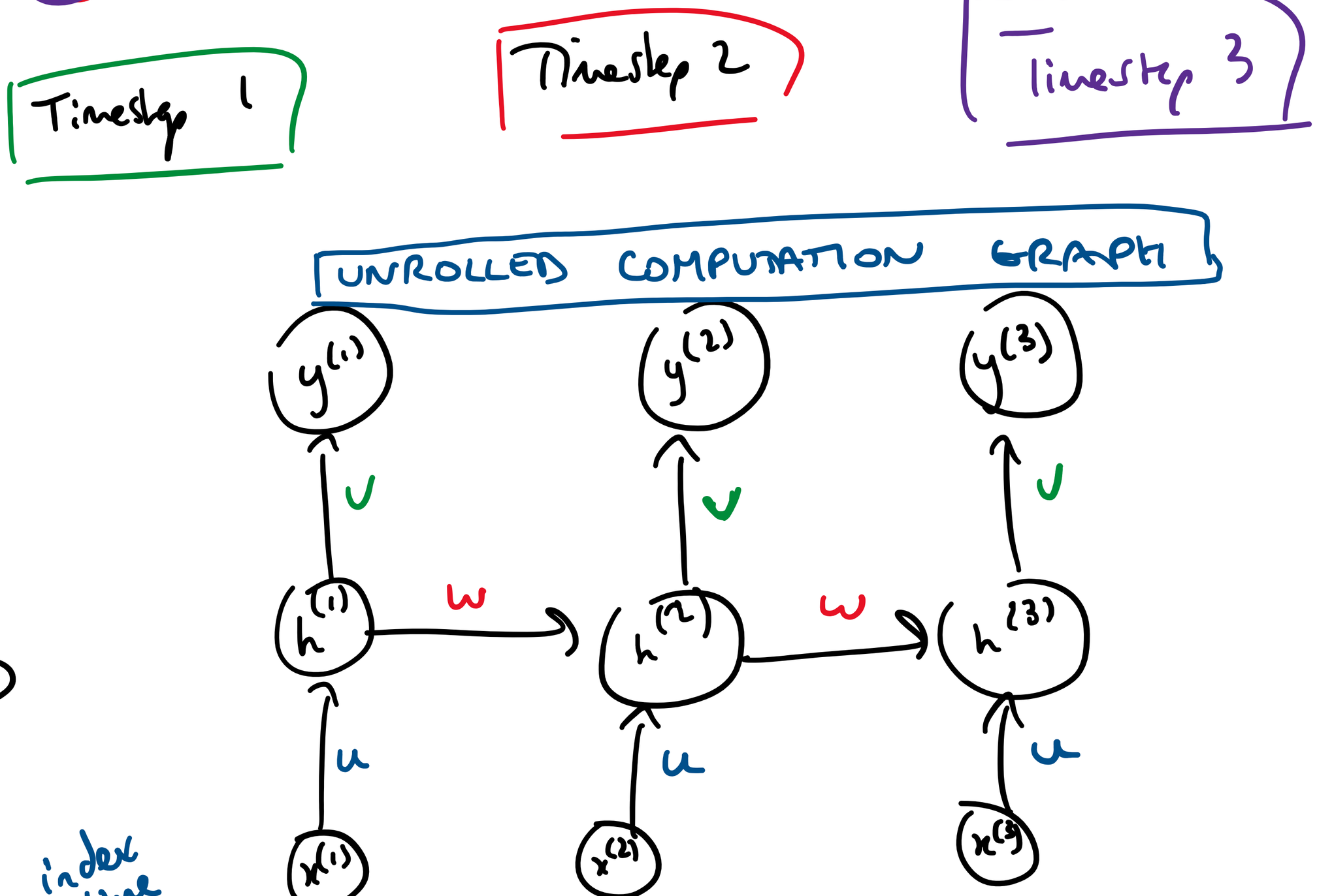
$3.5 = 1 + 1 - \frac{1}{2} + 2$

Example 2 : which of the 2 inputs has a largest sum

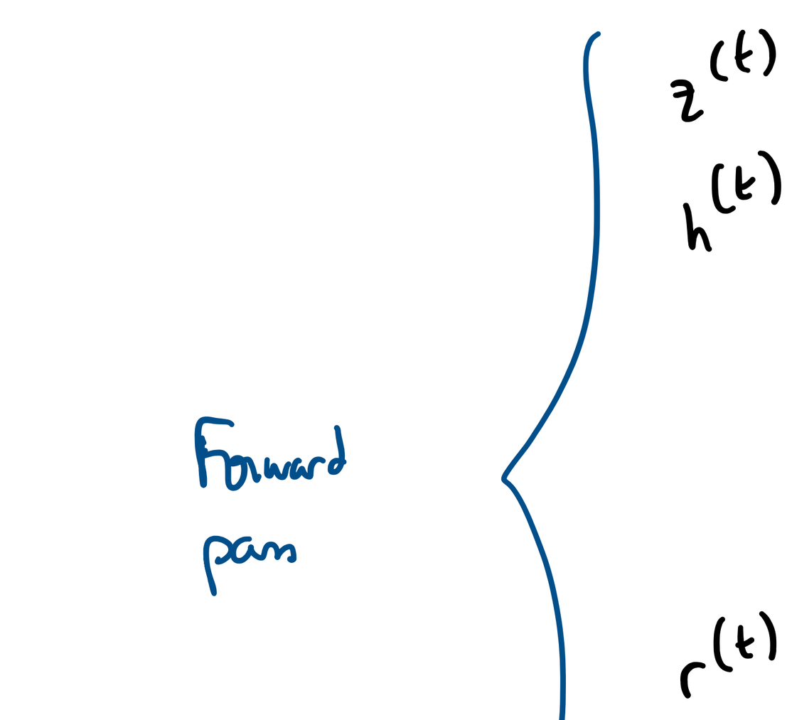


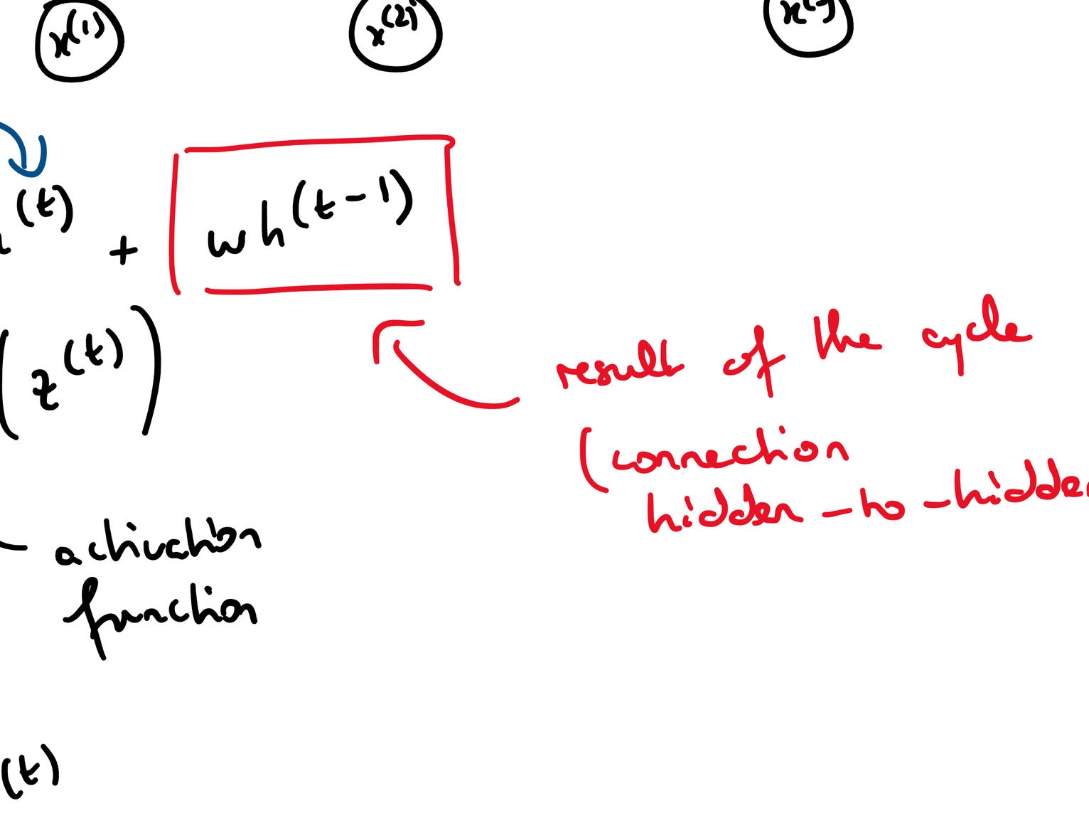
input 1 = (2, 0, 1)

input 2 = (-2, 3.5, 2.2)



Backpropagation through time





Forward pass

(index of time step)

$$z^{(t)} = u x^{(t)} + w h^{(t-1)}$$

$w h^{(t-1)}$: result of the cycle (connection hidden - to - hidden)

$$h^{(t)} = \phi(z^{(t)})$$

ϕ : activation function

$$r^{(t)} = v h^{(t)}$$

$$y^{(t)} = \phi(r^{(t)})$$

Backward pass

$$\overline{y^{(t)}} = \frac{\partial \mathcal{L}}{\partial y^{(t)}}$$

(we need to define the loss to derive this)

$$\overline{r^{(t)}} = \frac{\partial \mathcal{L}}{\partial r^{(t)}} = \frac{\partial \mathcal{L}}{\partial y^{(t)}} \frac{\partial y^{(t)}}{\partial r^{(t)}}$$

$$= \overline{y^{(t)}} \phi'(r^{(t)})$$

$$\overline{h^{(t)}} = \frac{\partial \mathcal{L}}{\partial h^{(t)}} = \frac{\partial \mathcal{L}}{\partial r^{(t)}} \frac{\partial r^{(t)}}{\partial h^{(t)}} + \frac{\partial \mathcal{L}}{\partial z^{(t+1)}} \frac{\partial z^{(t+1)}}{\partial h^{(t)}}$$

$$= \overline{r^{(t)}} v + \overline{z^{(t+1)}} w$$

$$\overline{z^{(t)}} = \frac{\partial \mathcal{L}}{\partial z^{(t)}} = \frac{\partial \mathcal{L}}{\partial h^{(t)}} \frac{\partial h^{(t)}}{\partial z^{(t)}}$$

$$= \overline{h^{(t)}} \phi'(z^{(t)})$$

T ← # of time steps

$$\overline{u} = \frac{\partial \mathcal{L}}{\partial u} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial z^{(t)}} \frac{\partial z^{(t)}}{\partial u}$$

$$= \sum_{t=1}^{T} \overline{z^{(t)}} x^{(t)}$$

$$\overline{v} = \frac{\partial \mathcal{L}}{\partial v} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial r^{(t)}} \frac{\partial r^{(t)}}{\partial v}$$

$$= \sum_{t=1}^{T} \overline{r^{(t)}} h^{(t)}$$

$$\overline{w} = \frac{\partial \mathcal{L}}{\partial w} = \sum_{t=2}^{T} \frac{\partial \mathcal{L}}{\partial z^{(t+1)}} \frac{\partial z^{(t+1)}}{\partial w}$$

$$= \sum_{t=2}^{T} \overline{z^{(t+1)}} h^{(t)}$$

Backprop through time = forward + backward pass

Only difference w/ fully connected / convolutional neural network is summation over time steps

Train RNNs w/ algorithms like stochastic gradient descent.

Cannot vectorize over time (because of summation)

Vectorize over training examples (minibatches) and units

⚠ RNNs often lead to vanishing / exploding gradients